CDS

TECHNICAL MEMORANDUM NO. CIT-CDS 97-009
June, 1997

"Robot Autonomous Navigation"

Xiaolin Feng and Wenhai Liu

Control and Dynamical Systems
California Institute of Technology
Pasadena, California 91125

Robot Autonomous Navigation

Xiaolin Feng
Wenhai Liu

California Institute of Technology
Pasadena, CA 91125

June 2nd, 1997

Contents

1 Introduction

Autonomous vehicle navigation is a very popular research area in the vision and control field. Based on Prof. Dickmanns' philosophy, we implement a navigation algorithm on the small robot. The robot can rely on its eyes (the camera mounted on the top of the robot) and control its wheels to walk through the sub-basement hallways of Caltech Moore Lab building. The speed we achieve is robot's mechanical maximum speed 0.5 m/s.

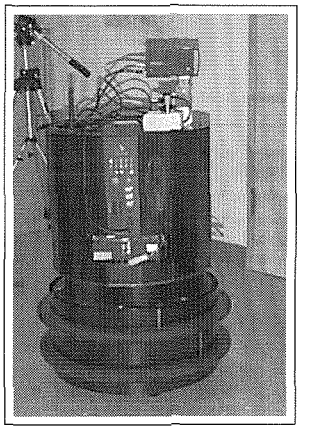


Figure 1: Nomad 200 system

2 Robot Hardware System

We use the robot Nomad 200 by NOMADIC TECH as our navigation vehicle. It is composed by three major parts: motor, sensory and computational system.

2.1 Motor System

The robot's mobile parts include three components: three wheels, one turret and one pan-tilt unit. All the motors are driven by 5 12-volts DC batteries.

Wheel System (2-DOF) The wheels are driven by two axes: translational and rotational. The three wheels are mechanically coupled to reduce slip in translation and always remain parallel to each other at any time while rotating.

Turret (1-DOF) The turret only rotates independently to the wheels. By this way, the robot can keep constant heading in navigation.

Pan-Tilt Unit (2-DOF) Pan-tilt unit is mounted on the top of turret and carries the camera. It has two motion axes, panting left and right and tilting up and down.

All the rotational motion are relative to a static base frame which provides the initial direction for these axes before the robot gets started walking.

2.2 Sensory Information

The robot has two sensory input: visual from a camera and tactile from a bumper sensor. Also, it can tell us its own status at any time.

2.2.1 Visual Camera Sensor

Visual sensor is the most important part in the project. It provides useful environment information to the robot. Based on this information , the robot can decide where to go for next step. The camera mounted on the top of pan-tilt unit is a Mitsubishi CCU-300 color camera. It provides a few zooming and focusing function, also a limited auto-gain control. Camera's output is digitized by DT3155 digital image acquisition board. The digitized image is black and white with 256 gray level per pixel.

2.2.2 Bumper Sensor

There is a total of 20 individual bumper sensors on the robot that are arranged in two rings of ten sensors each, one upper ring and one lower ring. These bumpers can effectively feel the contact from the outside environment and thus protect the robot.

2.2.3 Robot's Status

Information about the current status of the robot, its configuration and reading of the sensors can be obtained from a global STATE vector. The status is getting updated by a command routine. This provides useful information as a feedback for us to control the robot.

2.3 On-board Computational System

Robot's computational system consists of an on-board computer and a wireless communication setup. The main features are:

- One 120 MHz Pentium on a PCI backplane bus.
- 16 MB RAM
- Linux operating system
- Micro-controllers for motor control
- A speech synthesizer
- A wireless communication port for remote access to the robot.

3 Geometric Analysis

We first analyze the system from the geometric point of view. The robot cruises through the hallways by detecting the two stripes mounted as the intersecting line between the hallway walls and the floor. The stripes are parallel to each other. Also assume that the camera is parallel to the floor, i.e., there is no tilting angle. This can be archived by initializing the pan-tilt unit.

Let the origin of the world frame be on the hallway's centerline with x-axis perpendicular to the stripe edges and y parallel to the hallway direction.

Camera frame is created at the camera's image center. Assume the camera is locating H higher than the floor and d_x far away from the centerline. There is an angle offset ψ between the directions of camera's optical axis and hallway. (Geometry figure is shown in Figure 2.

Assume a point on the stripe edge has coordinate in the world frame: $P_w = (X_w, Y_w, 0)$, according to the geometric transformation between world

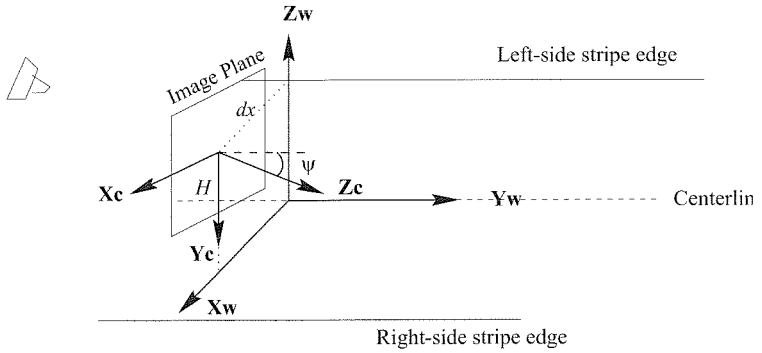


Figure 2: Geometric frame for the robot system

and camera frame :

$$P_c = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 \\ 0 & 0 & -1 \\ sin(\psi) & cos(\psi) & 0 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ 0 \end{bmatrix} + \begin{bmatrix} -d_x cos(\psi) \\ H \\ -d_x sin(\psi) \end{bmatrix} \tag{1}$$

and the pin-hole projection $p_c = (f\frac{P_c(1)}{P_c(3)}, f\frac{P_c(2)}{P_c(3)})$, its pixel coordinate on the image plane relative to the image center is:

$$p_c = \begin{bmatrix} f \times \frac{X_w cos(\psi) - Y_w sin(\psi) - d_x cos(\psi)}{X_w sin(\psi) + Y_w cos(\psi) - d_x sin(\psi)} \\ f \times \frac{H}{X_w sin(\psi) + Y_w cos(\psi) - d_x sin(\psi)} \end{bmatrix} \tag{2}$$

Let the vanishing point be the infinity intersection of two stripes on the image plane. Its pixel coordinate is easily obtained as: $p_{cv} = [-f \times tan(\psi), 0]$. Therefore, the camera's angle offset ψ can be calculated from detected vanishing point position. If given the edge pixel coordinate $p_c = (x_c, y_c)$ on camera image and X coordinate X_w from known hallway width, we can also derive the camera's distance offset by:

$$d_x = X_w - \frac{H}{y_c} \times (x_c \times cos(\psi) + f \times sin(\psi)) \tag{3}$$

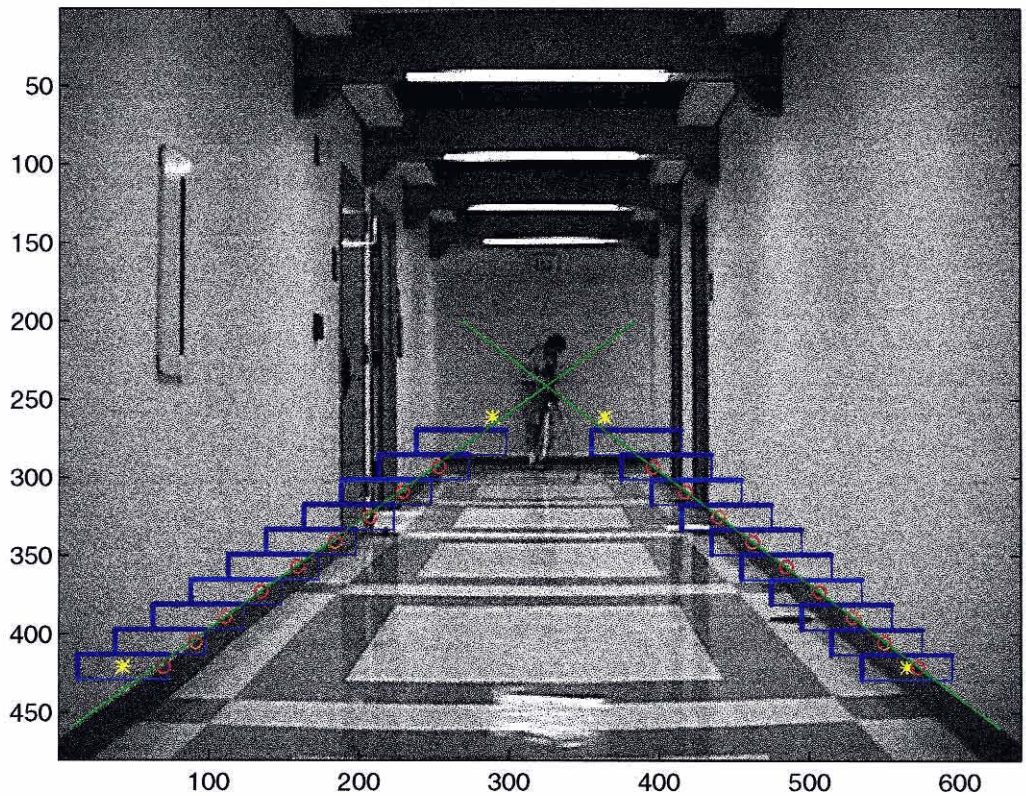


Figure 3: Typical image to show edge detection algorithm

4 Image Analysis

4.1 Edge Detection

In each frame, the computer needs to detect the stripe edges to obtain the angle offset. Our edge detection algorithm consists of searching in 20 small windows. The windows are opened along the predicted edge positions which were given by Kalman filter. In each window we find a most likely edge and finally linear-fit these data to get the whole stripes.

As shown in figure 3, the four yellow stars are points predicted for the position of edges and blue windows are opened to search small edges. To

avoid ending up with the edges of the floor pattern, we choose the most left or right possible edge in each window according to the corresponding side of wall. Also the edges we found should have enough gradient on both horizontal and vertical directions. This is because our camera doesn't tilt, so as long as it can see the stripes, the stripes must be neither horizontal nor vertical. By setting this threshold, the possible door edge can be discarded. The red circle points are what we found as edge points in each window. The position of the two edge intersection tells the location of vanishing point.

4.2 Focal Length Calibration

From the geometric analysis, we see the focal length is a critical variable we need to know. In the project, we develop a simple algorithm for focal length calibration. For each frame, the vanishing point (x_{cv}, y_{cv}) is at $(-f \times tan(\psi), 0)$. The robot can precisely control the rotation of the camera by controlling either turret or pan-tilt unit. We then take two images I1 and I2, where I2 is obtained by rotating $\Delta\psi$ with respect to I1. For both images, we have $x_{cv1} = -f \times tan(\psi_1)$ and $x_{cv2} = -f \times tan(\psi_1 + \Delta\psi)$. Since $x_{cv1}, x_{cv2}, \Delta\psi$ are known by edge detection and robot's rotation, the focal length f is calculated by solving the equation:

$$-tan(\Delta\psi)f^2 + (x_{cv1} - x_{cv2})f - x_{cv1}x_{cv2}tan(\Delta\psi) = 0 \qquad (4)$$

Figure 4 shows how much the vanishing point changes by rotating image (a) -5 degrees to get image (b).

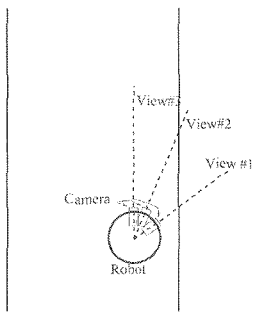
5 Autonomous Navigation

The robot walks through the hallways in the building. On each hallway, the algorithm consists of three phases: Initialization, Navigation, Turning at the end of hallway.

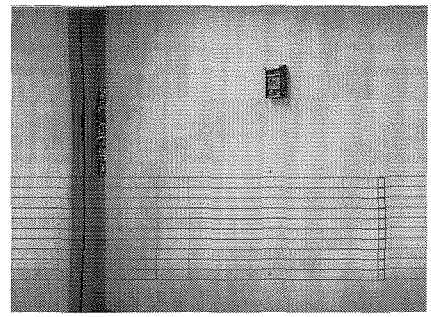
(a)

(b)

Figure 4: Vanishing point is changing with respective to camera's rotation



(a) top view of robot orientation



(b)view#1, no edge found

(c)view#2, one edge found

(d)view#3, facing to the hallway direction

Figure 5: Initialization

5.1 Initialization

The robot is located at an arbitrary position in the hallway. After hardware initialization, robot searches 10 width windows for possible left or right stripe edges. Turret is rotated 30 degree for each view until one or two edges are located. Extracting the angle difference between camera optical axis and the edges, robot rotates the camera to the direction of edge. After fine tuning of camera angle and extracting the left and right stripe edges, robot is in the initialized state, that camera and steering wheel faces to the hallway direction within ± 1 degree and distances to left and right walls are calculated.

5.2 Navigation

5.2.1 Navigation Diagram Flow

The initialization provides the first guess of robot's starting position. Then for each next step of navigation, the robot can predict its possible position by integrating the dynamic model. According to this prediction, we apply next step's control input. The reason we put control input here instead of waiting until we obtain the next step's true position is to avoid time delay of frame analysis. Following is to grab next frame. By detecting edges and analyzing the data, we can get the measurement of robot's position. Since we also have our prediction, but both prediction and measurement may have noise, Kalman filter combines two of them to give us best estimation of robot's position in this step. The diagram flow of this procedure is shown in figure 6:

5.2.2 Control Strategy

Our controllable state is d_x, the distance of the robot away from the centerline, and θ, the wheel angle offset from the centerline direction. (Notice that θ is obtained by ψ, the camera angle offset, and the angle difference of camera and wheel which can be read from robot status vector). Robot's System Model:

$$\begin{aligned} \dot{d_x} &= V \times sin(\theta) \approx V \times \theta \\ \dot{\theta} &= u \end{aligned} \tag{5}$$

Where: u: control input which is the angle velocity of wheel. V: wheel translation velocity. In order to enforce the robot walk along the centerline from arbitrary initial position, we design the PD controller:

$$u = -\frac{KP}{V} \times d_x - \frac{KD}{V} \times \dot{d_x} \tag{6}$$

The closed loop system function is:

$$\ddot{d_x} + KD \times \dot{d_x} + KP \times d_x = 0 \tag{7}$$

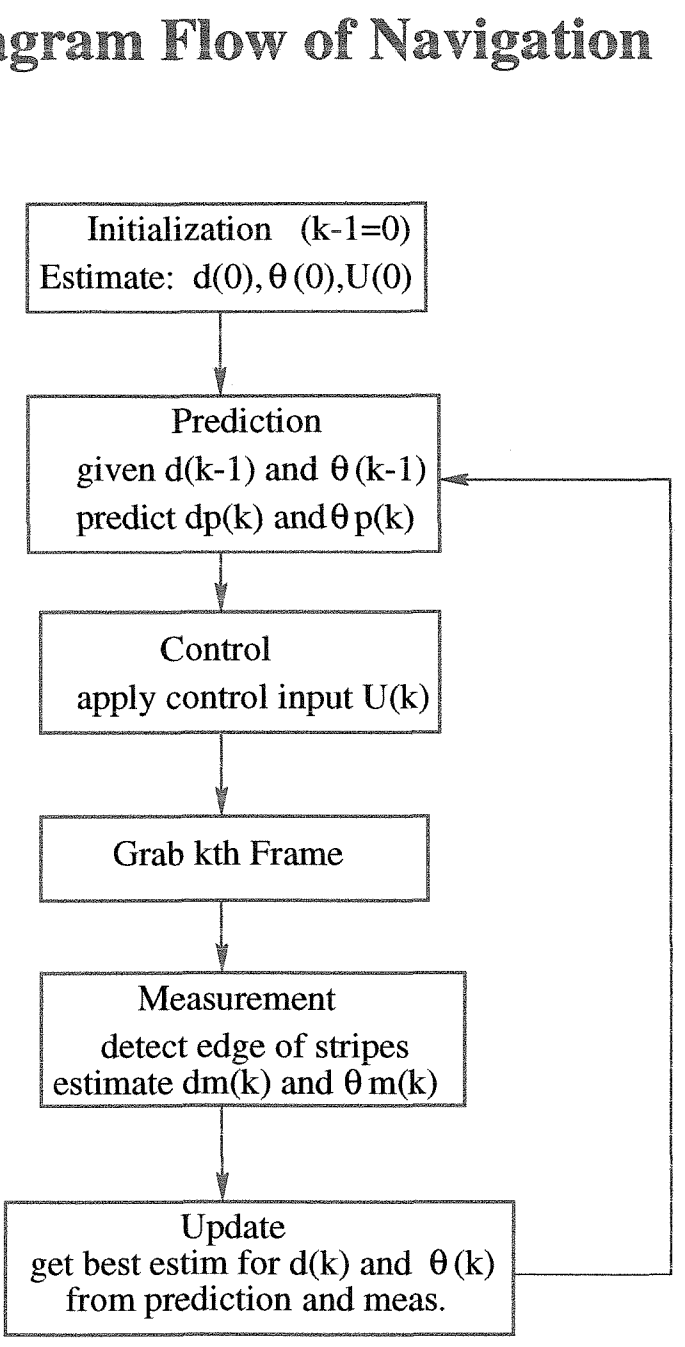
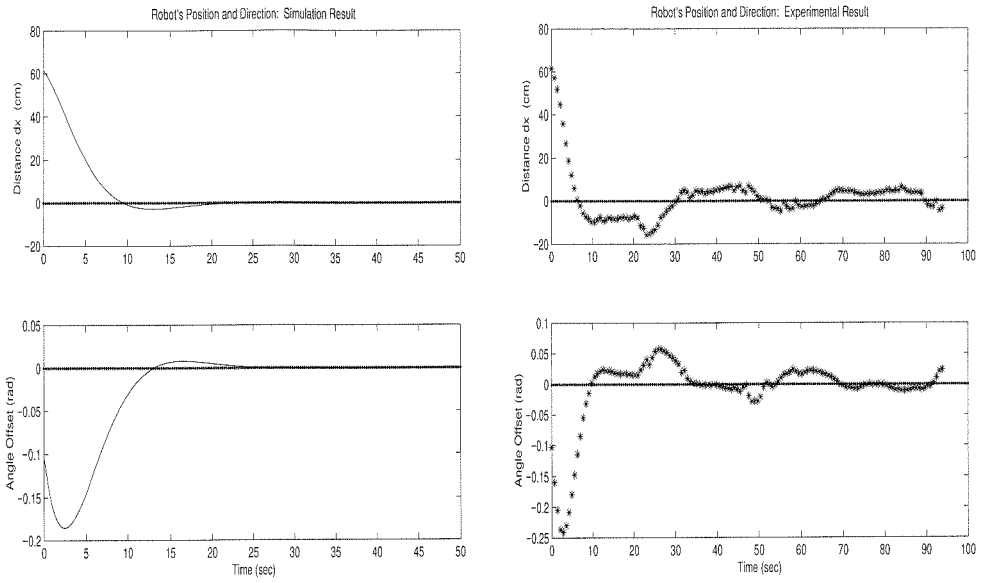


Figure 6: Flow diagram of navigation control



(a) (b)

Figure 7: Comparison between simulation (a) and experiment (b) results

- Stability Analysis: The system is stable if and only if KP and KD are positive.
- Performance Analysis: To get fast and best performance, we choose KP and KD such that the system's damping coefficient is 0.707.

The simulation and experimental results of the robot status under such control strategy are given in figure 7. The initial condition is $(d_x, \theta) = (61.4984, -0.1029)$.

5.2.3 Kalman Filter Estimation

Given the robot's dynamic model, and each step's status measurement from image analysis, we can use Kalman filter to get best estimation of the robot's true position. System model is given as (5), for numerical computation pur-

pose, assume u is constant in each step, we derive its discrete-time model:

$$\begin{bmatrix} d_x(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} 1 & V\Delta T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d_x(k) \\ \theta(k) \end{bmatrix} + \begin{bmatrix} \frac{1}{2}V\Delta T^2 \\ \Delta T \end{bmatrix} u(k) \qquad (8)$$

Discrete Kalman filter is easily applied to this model. We here skip the filter's formula which can be found in [1].

5.3 Turning At The Corner

When robot finally loses the stripe edges of the wall, we suppose that the robot has come to the end of the hallway within 5 meters (which is the case in all our experiments). Robot will search for the horizontal stripe edge of the facing wall within 2 to 10 meters in front of the robot. Once the front wall edge is found, the distance is calculated and robot moves straight ahead until 1.5 meters to the wall. Then turning is finished by rotate robot to the right by 90 degree. Robot is ready for navigate the new hallway.

For simplicity, the robot moves blindly to the front wall after extracting the distance to the wall. This is done with the assumption that robot is facing accurately to the wall and the distance is smaller than 10 meters. For our case, this simple algorithm works fine and safe. Since the control algorithm has the robot direction error smaller than 5 degree. Even robot walks blindly, it would not hit the wall as far as the distance is smaller than 10 meters. If the robot stops more than 10 meters away from the front wall, after turning 90 degree to right, the initialization algorithm will have the robot find the hallway direction and walk backward.

6 Experiment Performance

Above algorithm is implemented successfully on the robot. It can navigates around the sub-basement of Moore Lab with robot's maximum speed 0.5m/s from arbitrary position. The robot can automatically locating the direction of hallway, walking along the center line and turning to left/right at the end

of hallway. The movie of the robot navigating and the vision of robot are available.

7 Conclusion

Kalman filter and control algorithm are implemented onto a robot. The robot has been successfully navigating around the -known environment (hallway with known structure and width in our case).

For future works, it is straightforward to control robot locate a certain target upon navigation. Also it will be interesting to have the robot to learn and map an unknown or pseudo-unknown environment. For example, ask the robot to determine the width, length and structure of the hallway.

8 Acknowledge

We would like to thank Drs.Psaltis and Perona for their instruction and advises. Special thanks go to Greg, Mario,Bob and David, without their contribution and technical emergent service, the robot will never walk safely around the hallway.

References

[1] Arthur Gelb, Applied Optimal Estimation, 1974

[2] E.D.Dickmanns, An Integrated Spatio-Temporal Approach to Automatic Visual Guidance of Autonomous Vehicles, *IEEE transactions on systems*, Vol 20, No 6,1990

[3] Greg Steckman and George Ouyang, Robot Navigation, CNS 248 Report, 1996